Outlier Ensembles

[Position Paper]

Charu C. Aggarwal
IBM T. J. Watson Research Center
Yorktown Heights, NY
charu@us.ibm.com

ABSTRACT

Ensemble analysis is a widely used meta-algorithm for many data mining problems such as classification and clustering. Numerous ensemble-based algorithms have been proposed in the literature for these problems. Compared to the clustering and classification problems, ensemble analysis has been studied in a limited way in the outlier detection literature. In some cases, ensemble analysis techniques have been implicitly used by many outlier analysis algorithms, but the approach is often buried deep into the algorithm and not formally recognized as a general-purpose meta-algorithm. This is in spite of the fact that this problem is rather important in the context of outlier analysis. This paper discusses the various methods which are used in the literature for outlier ensembles and the general principles by which such analysis can be made more effective. A discussion is also provided on how outlier ensembles relate to the ensemble-techniques used commonly for other data mining problems.

1. INTRODUCTION

The outlier analysis problem has been widely studied by database, data mining, machine learning and statistical communities. Numerous algorithms have been proposed for this problem in recent years [3; 5; 11; 12; 26; 27; 21; 22; 35; 36]. A detailed survey on the topic may be found in [10], and a detailed book may be found in [1].

Data mining is often an inherently subjective process, where the objective function or model defined for a particular problem depends upon an analyst's understanding of the generative behavior of the data. Clearly, such assumptions are very subjective, and a specific algorithm being used may often model the underlying generative process in a limited way. In such cases, effective results can be obtained on some parts of the data which are modeled well, whereas the results on other parts of the data may not be very accurate. Similarly, a given model may sometimes behave well on a given data set, but may not behave well on other data sets. Ensemble analysis is a method which is commonly used in the literature in order to reduce the dependence of the model on the specific data set or data locality. This greatly increases the robustness of the data mining process. The ensemble technique is used very commonly in problems such as clustering and classification. Ensemble analysis is sometimes understood rather narrowly and referred to as the combination of the outputs of several independently executed data mining algorithms. A broader view of the term "ensemble analysis" can include any approach which combines the results of *either* dependent or independent executions of data mining algorithms. For example, the boosting technique in classification, in which the different executions of the classification algorithm are clearly dependent on one another, can also be considered an ensemble approach. The idea here is that the final result is an ensemble score from the results of different models, no matter how each of these models is derived.

The problem of ensemble analysis has been widely studied in the context of many data mining problems such as clustering and classification, though the approaches are different in supervised and unsupervised problems. In fact, each of these areas of meta-algorithm analysis is considered an active and vibrant subfield in its own right. To provide a specific example, the seminal paper [16] on boosting in classification has several thousand citations, and many different variants of the basic boosting approach have been proposed in the literature. The common methods used for ensemble analysis in clustering and classification are as follows:

- In clustering, the areas of alternative clustering, multiview clustering, and ensemble clustering are closely related subtopics of ensemble analysis. The idea in each of these variants is that the clustering process is inherently subjective, and a single clustering may not reflect the complete insights about how the data may cluster. Therefore, it is useful to examine the different and *alternative* clusters [7; 31; 32] and combine the results. Alternative clustering is also sometimes referred to as *multiview clustering*. The goal here is to determine clusterings which are significantly different from one another in order to obtain different insights. In some cases, the exploration of the clusters is performed visually [20] in order to obtain the best results.
- In the context of the classification problem, a variety of ensemble based methods have been proposed as as bagging [9], boosting [16], stacking [14; 38; 37], random forests [8], model averaging [15], and bucket of models [39]. Ensemble analysis is considered particularly important in noisy and streaming scenarios in which the quality of the results from individual classifiers is not considered robust because of limitations in data quality or processing time.

These different methods for ensemble analysis in clustering and classification enjoy wide popularity, and have been explored extensively in the literature. Furthermore, the different sub-topics (eg. bagging, boosting, etc.) in the ensemble analysis area are very well formalized.

This is remotely not true for outlier analysis, in which the work on ensemble analysis is rather patchy, sporadic, and not so well formalized. In many cases, useful meta-algorithms are buried deep into the algorithm, and not formally recognized as ensembles. Perhaps, one of the reasons why ensemble analysis has not been well explored in outlier analysis is that meta-algorithms require crisp evaluation criteria in order to show their relative merits over the base algorithm. Furthermore, evaluation criteria are often used in the intermediate steps of an ensemble algorithm (eg. boosting or stacking), in order to make future decisions about the precise construction of the ensemble. Among all core data mining problems, outlier analysis is the hardest to evaluate (especially on real data sets) because of a combination of its small sample space and unsupervised nature. The small sample space issue refers to the fact that a given data set may contain only a small number of outliers, and therefore the correctness of an approach is often hard to quantify in a statistically robust way. This is also a problem for making robust decisions about future steps of the algorithm, without causing over-fitting. The unsupervised nature of the problem refers to the fact that no ground truth is available in order to evaluate the quality of a component in the ensemble. This necessitates the construction of simpler ensembles with fewer qualitative decisions about the choice of the components in the ensemble. These factors have been a significant impediment in the development of effective meta-algorithms. On the other hand, since the classification problem has the most crisply defined criteria for evaluation, it also has the richest meta-algorithm literature among all data mining problems. This is because the problem of model evaluation is closely related to quality-driven meta-algorithm development.

Nevertheless, a number of examples do exist in the literature for outlier ensembles. These cases show that when ensemble analysis is used properly, the potential for algorithmic improvement is significant. Ensemble analysis has been used particularly effectively in high-dimensional outlier detection [18; 24; 28; 30; 32; 33], in which multiple subspaces of the data are often explored in order to discover outliers. In fact, the earliest *formalization* [28] of outlier ensemble analysis finds its origins in high dimensional outlier detection, though informal methods for ensemble analysis were proposed much earlier to this work. The high dimensional scenario is an important one for ensemble analysis, because the outlier behavior of a data point in high dimensional space is often described by a subset of dimensions, which are rather hard to discover in real settings. In fact, most methods for localizing the subsets of dimensions can be considered *weak guesses* to the true subsets of dimensions which are relevant for outlier analysis. The use of multiple models (corresponding to different subsets of dimensions) reduces the uncertainty arising from an inherently difficult subspace selection process, and provides greater robustness for the approach. The feature bagging work discussed in [28] may be considered a first *formal* description of outlier ensemble analysis in a real setting. However, as we will see in this article, numerous methods were proposed earlier to this work which could be considered ensembles, but were never formally recognized as ensembles in the literature. As noted in [18], even the first high dimensional outlier detection approach [3] may be considered an ensemble method, though it was not formally presented as an ensemble method in the original paper. It should also be pointed out that while high dimensional data is an important case for ensemble analysis, the potential of ensemble analysis is much broader, and is likely to apply to any scenario in which outliers are defined from varying causes of rarity. Furthermore, many types of ensembles such as sequential ensembles can be used in order to successively refine data-centric insights.

This paper will discuss the different methods for outlier ensemble analysis in the literature. We will provide a classification of the different kinds of ensembles, and the key parts of the algorithmic design of ensembles. The specific importance of different parts of algorithmic design will also be discussed. Ensemble algorithms can be categorized in two different ways:

- *Categorization by Component Independence:* Are the different components of the ensemble independent of one another or do they depend on one another? To provide an analogy with the classification problem, boosting can be considered a problem in which the different components of the ensemble are not independent of one another. This is because the execution of a specific component depends upon the result from previous executions. On the other hand, many forms of classification ensembles such as bagging are those in which the classification models are independent of one another.

- *Categorization by Component Type:* Each component of an ensemble can be defined on the basis of either *data choice* or *model choice*. The idea in the former is to carefully pick a subset of the data or data dimensions (eg. boosting/bagging in classification) or to pick a specific algorithm (eg. stacking or model-ensembles). The categorization by component type is related to categorization by component independence, because data-centered ensembles are often sequential, whereas model-centered ensembles are often independent. However this is not always the case. For example, independent data centered bagging methods [28] are often used in outlier analysis.

It should be pointed out that the aforementioned categorizations of different kinds of ensembles are inherently incomplete, and it is impossible to fully describe every possibility. For example, it is possible for the different components to be heterogeneous, which are defined on the basis of different aspects of the data and models [34]. However, such models are less frequent in the outlier analysis literature, because of the complexity of reasonably evaluating the importance of different ensemble components.

A typical outlier ensemble contains a number of different components, which are used to construct the final result.

- *Model Creation:* This is the individual methodology or algorithm which is used to create the corresponding component of the ensemble. In some cases, the methodology may be simply that of random subspace sampling.

- *Normalization:* Different methods may create outlier scores which are on very different scales. In some cases, the scores may be in ascending order, whereas in others, they may be in descending order. In such cases, normalization is important in being able to combine the scores meaningfully, so that the outlier scores from different components are roughly comparable.

- *Model Combination:* This refers to the final combination function, which is used in order to create the outlier score.

This paper is organized as follows. The next section will discuss the categorization of ensembles on the basis of component independence. Section 3 will discuss the categorization of ensembles on the basis of model type. Section 4 will study the role of the combination function in different kinds of ensemble analysis. Section 5 discusses meta-algorithms for other data mining problems in the literature, and whether such ideas can be adapted to the outlier analysis scenario. Section 6 contains the conclusions and summary.

2. CATEGORIZATION BY COMPONENT INDEPENDENCE

This categorization examines whether the components are developed independently, or whether they depend on one another. There are two primary kinds of ensembles, which can be used in order to improve the quality of outlier detection algorithms:

- In *sequential ensembles*, a given algorithm or set of algorithms are applied sequentially, so that future applications of the algorithms are impacted by previous applications, in terms of either modifications of the base data for analysis or in terms of the specific choices of the algorithms. The final result is either a weighted combination of, or the final result of the last application of an outlier analysis algorithm. For example, in the context of the classification problem, boosting methods may be considered examples of sequential ensembles.

- In *independent ensembles*, different algorithms, or different instantiations of the same algorithm are applied to either the complete data or portions of the data. The choices made about the data and algorithms applied are independent of the results obtained from these different algorithmic executions. The results from the different algorithm executions are combined together in order to obtain more robust outliers.

In this section, both kinds of ensembles will be studied in detail.

2.1 Sequential Ensembles

In sequential-ensembles, one or more outlier detection algorithms are applied sequentially to either all or portions of the data. The core principle of the approach is that each application of the algorithms provides a better understanding of the data, so as to enable a more refined execution with either a modified algorithm or data set. Thus, depending upon the approach, either the data set or the algorithm may be changed in sequential executions. If desired, this approach can either be applied for a fixed number of times, or be used in order to converge to a more robust solution. The broad framework of a sequential-ensemble algorithm is provided in Figure 1.

```
Algorithm SequentialEnsemble(Data Set: $\mathcal{D}$
      Base Algorithms: $\mathcal{A}_1 \ldots \mathcal{A}_r$)
begin
  $j = 1$;
  repeat
    Pick an algorithm $\mathcal{A}_j$ based on results from
        past executions;
    Create a new data set $f_j(\mathcal{D})$ from $\mathcal{D}$ based
        on results from past executions;
    Apply $\mathcal{A}_j$ to $f_j(\mathcal{D})$;
    $j = j + 1$;
  until(termination);
  report outliers based on combinations of results
      from previous executions;
end
```

Figure 1: Sequential Ensemble Framework

In each iteration, a successively refined algorithm may be used on a refined data, based on the results from previous executions. The function $f_j(\cdot)$ is used to create a refinement of the data, which could correspond to data subset selection, attribute-subset selection, or generic data transformation methods. The description above is provided in a very general form, and many special cases can be possibly instantiated from this general framework. For example, in practice, only a single algorithm may be used on successive modifications of the data, as data is refined over time. Furthermore, the sequential ensemble may be applied in only a small number of constant passes, rather than a generic convergence-based approach, as presented above. The broad principle of sequential ensembles is that a greater knowledge of data with successive algorithmic execution helps focus on techniques and portions of the data which can provide fresh insights.

Sequential ensembles have not been sufficiently explored in the outlier analysis literature as general purpose meta-algorithms. However, many *specific* techniques in the outlier literature use methods, which can be recognized as special cases of sequential ensembles. A classic example of this is the use of two-phase algorithms for building a model of the normal data. In the first-phase, an outlier detection algorithm is used in order to remove the obvious outliers. In the second phase, *a more robust* normal model is constructed after removing these obvious outliers. Thus, the outlier analysis in the second stage is much more refined and accurate. Such approaches are commonly used for cluster-based outlier analysis (for constructing more robust clusters in later stages) [6], or for more robust histogram construction and density estimation. However, most of these methods are presented in the outlier analysis literature as specific optimizations of *particular* algorithms, rather than as general meta-algorithms which can improve the effectiveness of an *arbitrary* outlier detection algorithm. There is significant scope for further research in the outlier analysis literature, by recognizing these methods as general-purpose ensembles, and using them to improve the effectiveness of outlier detection. In these models, the goal of the sequential ensemble is data refinement. Therefore, the score returned by the last stages of the ensemble is the most relevant outlier score.

```
Algorithm IndependentEnsemble(Data Set: $\mathcal{D}$
        Base Algorithms: $\mathcal{A}_1 \ldots \mathcal{A}_r$)
begin
  $j = 1$;
  repeat
    Pick an algorithm $\mathcal{A}_j$;
    Create a new data set $f_j(\mathcal{D})$ from $\mathcal{D}$;
    Apply $\mathcal{A}_j$ to $f_j(\mathcal{D})$;
    $j = j + 1$;
  until(termination);
  report outliers based on combinations of results
      from previous executions;
end
```

Figure 2: Independent Ensemble Framework

Another example of a sequential ensemble is proposed in [30] in which different subspaces of the data are *recursively* explored, on the basis of their discriminative behavior. A subspace is explored only if one of its predecessor[1] subspaces is also sufficiently discriminative. Thus, this approach is sequential, since the construction of future models of the ensemble is dependent on the previous models. The goal of the sequential ensemble is the discovery of other related subspaces which are also discriminative. Nevertheless, since the sequential approach is combined with enumerative exploration of different subspace extensions, the combination function in this case needs to include the scores from the different subspaces in order to create an outlier score. The work in [30] uses the product of the outlier scores of the discriminative subspaces as the final result. This is equivalent to using an aggregate on the logarithmic function of the outlier score.

2.2 Independent Ensembles

In independent ensembles, different instantiations of the algorithm or different portions of the data are used for outlier analysis. Alternatively, the same algorithm may be applied, but with either a different initialization, parameter set or even random seed in the case of a randomized algorithms. The results from these different algorithm executions can be combined in order to obtain a more robust outlier score. A general purpose description of independent ensemble algorithms is provided in the pseudo-code description of Figure 2.

The broad principle of independent ensembles is that different ways of looking at the same problem provides more robust results which are not dependent on specific artifacts of a particular algorithm or data set. Independent ensembles have been explored much more widely and formally in the outlier analysis literature, as compared to sequential ensembles. Independent ensembles are particularly popular for outlier analysis in high-dimensional data sets, because they enable the exploration of different subspaces of the data in which different kinds of deviants may be found.

Examples exist of both picking different algorithms and data sets, in order to combine the results from different executions. For example, the methods in [28; 29] sample subspaces from the underlying data in order to determine outliers from each of these executions independently. Then, the results from these different executions are combined in order to determine the outliers. The idea in these methods is that results from different subsets of sampled features may be bagged in order to provide more robust results. Some of the recent methods for subspace outlier ranking and outlier evaluation can be considered independent ensembles which combine the outliers discovered in different subspaces in order to provide more robust insights.

[1]A predecessor is defined as a subspace with one dimension removed.

3. CATEGORIZATION BY CONSTITUENT COMPONENTS

In general, a particular component of the model may use a different model, and a different subset or subspace of the data [34]. However, this is rarely done in practice. Typically, each component of the model is either defined as a specific model, or as a specific part of the data. The former type of ensemble is referred to as *model-centered*, whereas the latter type is referred to as *data-centered*. Each of these specific types will be discussed in detail in this section.

3.1 Model-centered Ensembles

Model centered ensembles attempt to combine the outlier scores from different models built on the same data set. The major challenge of this model is that the scores from different models are often not directly comparable to one another. For example, the outlier score from a k-nearest neighbor approach is very different from the outlier score provided by a PCA-based detection model. This causes issues in combining the scores from these different outlier models. Therefore, it is critical to be able to convert the different outlier scores into normalized values which are directly comparable, and also preferably interpretable, such as a probability [17]. This issue will be discussed in the next section on defining combination functions for outlier analysis. Another key challenge is in terms of the specific definition of the combination function for outliers. Should we use model averaging, best fit or worst fit? This problem is of course not specific to model-centered ensembles.

A particular form of model-centered ensembles which are commonly used in outlier analysis, but not formally recognized as ensembles is the issue of using the same model over different choices of the underlying model parameters, and then combining the scores. This is done quite frequently in many classical outlier analysis algorithms such as LOCI [35] and LOF [12]. However, since the approach is interpreted as a question of parameter tuning, it is not recognized formally as an ensemble. In reality, any systematic approach for parameter tuning, *which is dependent on the output scores and directly combines or uses the outputs of the different executions* should be interpreted as an ensemblar approach. This is the case with the LOF and LOCI methods. Specifically, the following ensemblar approach is used in the two methods.

- In the LOF method, the model is run over a range of values of k, which defines the neighborhood of the data points. The work in [12] examines the use of different combination functions such as the minimum, average or the maximum of the LOF values as the outlier score. It is argued in [12], that the appropriate combination function is to use the *maximum* value in order to prevent dilution of the outlier scores by inappropriate parameter choices in the model. In other words, the specific model which *best enhances* the outlier behavior for a data point is used.

- The LOCI method uses a multi-granularity approach, which uses a sampling neighborhood in order to determine the level of granularity in which to compute the outlier score. Different sampling neighborhoods are used, and a point is declared as an outlier based on the neighborhood in which its outlier behavior is *most* enhanced. It is interesting to note that the LOCI method uses a very similar combination function as the LOF method in terms of picking the component of the ensemble which *most* enhances the outlier behavior.

It should be pointed out that when the different components of the ensemble create comparable scores (eg. different runs of a particular algorithm such as LOF or LOCI), then the combination process is greatly simplified, since the scores across different components are comparable. However, this is not the case, when the different components create scores which are not directly comparable to one another. This issue will be discussed in a later section on defining combination functions.

3.2 Data-centered Ensembles

In data-centered ensembles, different parts, samples or functions of the data are explored in order to perform the analysis. It should be pointed out that a function of the data could include either a sample of the data (horizontal sample) or a relevant subspace (vertical sample). More general functions of the data are also possible, though have rarely been explored in the literature. The core idea is that each part of the data provides a specific kind of insight, and by using an ensemble over different portions of the data, it is possible to obtain different insights.

One of the earliest data-centered ensembles was discussed in [28]. In this approach, random subspaces of the data are sampled, and the outliers are determined in these projected subspaces. The final outliers are declared as a combination function of the outliers from the different subspaces. This technique is also referred to as the *feature bagging* or *subspace ensemble* method. The core algorithm discussed in [28] is as follows:

```
Algorithm FeatureBagging(Data Set $\mathcal{D}$);
begin
  repeat
  Sample a subspace between $d/2$ and $d$ dimensions;
  find $LOF$ score for each point in projected representation;
  until $n$ iterations;
  Report combined scores from different subspaces;
end
```

Two different methods are used for combining scores. The first uses the best rank of a data point in any projection in order to create the ordering. A variety of methods can be used for tie breaking. The second method averages the scores over the different executions. Another method discussed in [17] converts the outlier scores into probabilities before performing the bagging. This normalizes the scores, and improves the quality of the final combination.

A number of techniques have also been proposed for statistical selection of relevant subspaces for ensemble analysis [24; 30]. The work in [30] determines subspaces which are relevant to each data point. The approach is designed in such a way, that For the discriminative subspaces found by the method, the approach uses the *product* of (or the addition of the logarithm of) the outlier scores in the different discriminative subspaces. This can be viewed as a combination of model averaging and selection of the most discriminative subspaces, when the scores are scaled by the logarithmic function. The work in [24] is much closer to the feature bagging method of [28], except that statistical selection of relevant subspaces is used for the outlier analysis process. The final score is computed as the average of the scores over different components of the ensemble. Recently, a method called *OutRank* [33] has been proposed, which can combine the results of multiple rankings based on the relationship of data points to their nearest subspace clusters. It has been shown that even traditional subspace clustering algorithms [4] can provide good results for outlier analysis, when the ensemble method is used. This, the work in [33] conclusively shown the power of ensemble analysis for high dimensional data.

A different data-centered ensemble which is commonly used in the literature, but often not recognized as an ensemblar approach is that of using initial phases of removing outliers from a data set, in order to create a more refined model for outlier analysis. An example of such an approach in the context of intrusion detection is discussed in [6]. In these cases, the combination function can be simply defined as the result from the very last step of the execution. This is because the data quality is improved significantly from the early components of the ensemble, and the results in the last phase reflect the outliers most accurately. This is because this is also a sequential ensemble with a specific goal of data refinement.

It should be pointed out that the distinction in this section between model-centered and data-centered ensembles is a somewhat semantic one, since a data-centered ensemble can also be considered a specific type of model-centered ensemble. Nevertheless, this categorization is useful, because the exploration of different segments of the data requires inherently different kinds of techniques than the exploration of different models which are data-independent. The choices in picking different functions of the data for exploration requires data-centric insights, which are analogous to classification methods such as boosting, especially in the sequential case. Therefore, we view this categorization as a convenient way to stimulate different lines of research on the topic.

3.3 Discussion of Categorization Schemes

The two different categorization schemes are clearly not exhaustive, though they represent a significant fraction of the ensemble functions used in the literature. In fact, these two categorization schemes can be combined in order to create four different possibilities. This is summarized in Table 1. We have also illustrated how many of the current ensemblar schemes map to these different possibilities. Interestingly, we were unable to find an example of a sequential model-based ensemble in the literature, though it is possible that the results from the execution of a particular model can provide hints about future directions of model construction for an outlier analysis algorithm. Therefore, it has

	Data Centered	Model Centered
Indep.	Feature Bagging [28]	LOF Tuning [12]
	HICS [24]	LOCI Tuning [35]
	Multiple Proclus [17]	Isolation For. [29]
	OutRank [33]	
	Nguyen et al [34]	
	Converting scores into probabilities [25]	
	Calibrated Bagging [17]	
Seq.	Intrusion Bootstrap [6]	Open
	OUTRES [30]	

Table 1: Categorization of Ensemble Techniques

been classified as an open problem in our categorization, and would be an interesting avenue for future exploration. The work by Nguyen et al [34] cannot be classified as either a data-centered or a model-centered scheme, since it uses some aspects of both. Furthermore, the work in [17; 25] convert outlier scores into probabilities as a general preprocessing method for normalization, and are not dependent on whether the individual components are data-centered or model-centered. The issue of model combination is a critically tricky one both in terms of how the individual scores are normalized, and in terms of how they are combined. This issue will be discussed in detail in the next section.

4. DEFINING COMBINATION FUNCTIONS

A crucial issue in outlier analysis is the definition of combination functions which can combine the outlier scores from different models. There are several challenges which arise in the combination process:

- *Normalization Issues:* The different models may output scores which are not easily comparable with one another. For example, a k-nearest neighbor classifier may output a distance score, which is different from an LOF score, and the latter is also quite different from the MDEF score returned by the LOCI method. Even a feature bagging approach, which is defined with the use of the same base algorithm (LOF) on different feature subsets, may sometimes have calibration issues in the scores [17]. Therefore, if a combination function such as the average or the max is applied to the constituent scores, then one or more the models may be inadvertently favored.
- *Combination Issues:* The second issue is the choice of the combination function. Given a set of normalized outlier scores, how do we decide the specific choice of the combination function to be used? Should the minimum of the scores be used, the average of the scores be used, or the maximum of the scores be used. It turns out that the answer to this question may sometimes depend on the specific constituent components of the model, though it would seem that some choices are more common than others in the literature.

In the following section, we will discuss some of these issues in detail.

4.1 Normalization Issues

The major factor in normalization is that the different algorithms do not use the same scales of reference and cannot be reasonably compared with one another. In fact, in some cases, high outlier scores may correspond to larger outlier tendency, whereas in other cases, low scores may correspond to greater outlier tendency. This causes problems during the combination process, since one or more components may be inadvertently favored. One simple approach for performing the normalization is to use the ranks from the different outlier analysis algorithms from greatest outlier tendency to least outlier tendency. These ranks can then be combined in order to create a unified outlier score. One of the earliest methods for feature bagging [28] uses such an approach in one of its combination functions.

The major issue with such an approach is that it does lose a lot of information about the relative differences between the outlier scores. For example, consider the cases where the top outlier scores for components A and B of the ensemble are $\{1.71, 1.71, 1.70\ldots\}$ and $\{1.72, 1.03, 1.01\ldots\}$ respectively, and each component uses (some variation of) the LOF algorithm. It is clear that in component A, the top three outlier scores are almost equivalent, and in component B, the top outlier score is the most relevant one. However, a ranking approach will not distinguish between these scenarios, and provide them the same rank values. Clearly, this loss of information is not desirable for creating an effective combination from the different scores.

The previous example suggests that it is important to examine both the ordering of the values and the distribution of the values during the normalization process. Ideally, it is desirable to somehow convert the outlier scores into probabilities, so that they can be reasonably used in an effective way. An approach was proposed in [17] which uses mixture modeling in conjunction with the EM-framework in order to convert the scores into probabilities. Two methods are proposed in this work. Both of these techniques use parametric modeling methods. The first method assumes that the posterior probabilities follow a logistic sigmoid function. The underlying parameters are then learned from the EM framework from the distribution of outlier scores. The second approach recognizes the fact that the outlier scores of data points in the outlier component of the mixture is likely to show a different distribution (Gaussian distribution), than the scores of data points in the normal class (Exponential distribution). Therefore, this approach models the score distributions as a mixture of exponential and Gaussian probability functions. As before, the parameters are learned with the use of the EM-framework. The posterior probabilities are calculated with the use of the Bayes rule. This approach has been shown to be effective in improving the quality of the ensemble approach proposed in [28]. A second method has also been proposed recently [25], which improves upon this base method for converting the outlier scores, and converting the scores into probabilities.

4.2 Combining Scores from Different Models

The second issue is the choice of the function which needs to be used in order to combine the scores. Given a set of r (normalized) outlier scores $Score_i(\overline{X})$ for the data point $\overline{X}$, should we use the model average, maximum, or minimum? For ease in discussion in this section, we will assume the convention without loss of generality that greater outlier scores correspond to greater outlier tendency. Therefore the maximum function picks the worst fit, whereas the minimum function picks the best fit.

The earliest work on ensemble-based outlier analysis (not formally recognized as ensemble analysis) was performed in the context of model parameter tuning [12; 35]. Most outlier analysis methods typically have a parameter, which controls the *granularity* of the underlying model. The outliers may often be visible to the algorithm only at a specific level of granularity. For example, the value of k in the k-nearest neighbor approach or LOF approach, the sampling neighborhood size in the LOCI approach, the number of clusters in a clustering approach all control the granularity of the analysis. What is the optimal granularity to be used? While this is often viewed as an issue of parameter tuning, it can also be viewed as an issue of ensemble analysis, when addressed in a certain way.

In particular, the methods in [12; 35] run the algorithms over a range of values of the granularity parameter, and pick the parameter choice which best enhances the outlier score (maximum function for our convention on score ordering) for a given data point. In other words, we have:

$$Ensemble(\overline{X}) = \text{MAX}_i\{Score_i(\overline{X})\} \qquad (1)$$

This reason for this has been discussed in some detail in the original LOF paper. In particular, it has been suggested that the use of other combination function such as the average or the minimum leads to a dilution in the outlier scores from the irrelevant models. This seems to be a reasonable choice at least from an intuitive perspective.

Some other common functions which are used in the literature are as follows:

- *Maximum Function:* This is one of the most common functions used for combining ensemblar scores both in implicit (LOF and LOCI parameter tuning) and explicit ensemblar models. One variation on this model is to use the ranks instead of the scores in the combination process. Such an approach was also used in feature bagging [28]. An important aspect of the process is that the different data points need to have the same number of components in the ensemble in order to be compared meaningfully.

- *Averaging Function:* In this case, the model scores are averaged over the different components of the ensemble. The risk factor here is that if the individual components of the ensemble are poorly derived models, then the irrelevant scores from many different components will dilute the overall outlier score. Nevertheless, such an approach has been used extensively. Examples of methods which use this method are one of the models in feature bagging [28], the HICS method [30], and a recent approach described in [25].

- *Damped Averaging:* In this model, a damping function is applied to the outlier scores before averaging, in order to prevent it from being dominated by a few components. Examples of a damping function could be the square root or the logarithm. It should be pointed out that the use of the product of the outlier scores or geometric averaging could be interpreted as the averaging of the logarithm of outlier scores.

- *Pruned Averaging and Aggregates:* In this case, the low scores are pruned and the outlier scores are either averaged or aggregated (summed up) over the relevant ensembles. The goal here is to prune the irrelevant models for each data point before computing the combination score. The pruning can be performed by either using an absolute threshold on the outlier score, or by picking the top k models for each data point, and averaging them. The risk factor in using absolute thresholds are the normalization issues which arise from different data points having different ensemblar components. Both the average and aggregate scores can no longer be meaningfully compared across different data points. Aggregates are more appropriate than averages, since they implicitly count the number of ensemble components in which a data point is relevant. A data point will be more relevant in a greater number of ensemble components, when it has a greater tendency to be an outlier.

- *Result from Last Component Executed:* This approach is sometimes used in sequential ensembles [6], in which each component of the ensemble successively refines the data set, and removes the obvious outliers. As a result, the normal model is constructed on a data set from which outliers are removed and the model is more robust. In such cases, the goal of each component of the sequential ensemble is to successively refine the data set. Therefore, the score from the last component is the most appropriate one to be used.

Which combination function provides the best insights for ensemble analysis? Clearly, the combination function may be dependent on the structure of the ensemble in the general case, especially if the function of each component of the ensemble is to either refine the data set, or understand the behavior of only a very *local* segment of the data set.

However, for the general case, in which the function of each component of the ensemble is to provide a reasonable and comparable outlier score for each data point, the two most commonly used functions are the *Maximum* and the *Averaging* functions. While pruned averaging combines these aspects, it is rarely used in ensemble analysis. Which combination function is best? Are there any other combination functions which could conceivably provide better results? These are open questions, the answer to which is not completely known because of the sparse literature on outlier ensemble analysis. It is this author's personal opinion, that the intuitive argument provided in the LOF paper [12] on using the maximum function for avoiding dilution from irrelevant models is the correct one in many scenarios. However, the issue is certainly not settled in the general case, and many variants such as pruned averaging may also provide robust results, while avoiding most of the irrelevant models.

5. POSSIBLE AVENUES OF EXPLORATION: LEARNING FROM OTHER DATA MINING PROBLEMS

The area of outlier ensemble analysis is still in its infancy, though it is rapidly emerging as an important area of research in its own right. Currently, the diversity of algorithms available for outlier ensemble analysis is limited, and is nowhere close to many other data mining problems such as clustering and classification. Therefore, it may be instructive to examine some of the key techniques for other data mining problems such as clustering or classification, and whether it makes sense or is even feasible to design analogous methods for the outlier analysis problem.

As discussed earlier, a major challenge for ensemble development in unsupervised problems is that the evaluation process is highly subjective, and therefore the quality of the intermediate results cannot be fully evaluated in many scenarios. One of the constraints is that the intermediate decisions must be made with the use of outlier scores only, rather than with the use of concrete evaluation criteria on hold-out sets (as in the case of the classification problem). Therefore, in this context, we believe that the major similarities and differences in supervised and unsupervised methods are as follows:

- **Intermediate Evaluation:** In unsupervised methods, ground truth is typically not available. While one can use measures such as classification accuracy in supervised methods, this is not the case with unsupervised methods. Intermediate evaluation is particularly important for sequential methods. This is one of the reasons that sequential methods are much rarer than independent methods in outlier ensemble analysis.

- **Diversity and Consensus Issues:** Both supervised and unsupervised methods seek greater diversity with the use of an ensemble in terms of the methodology used for creating the model. In many cases, this is done by selecting models which are different from one another. For example, in clustering, diversity is achieved by using either randomized clustering or explicitly picking orthogonal clusterings [2]. However, in the case of supervised methods, the level of consensus is also measured *at the end* in terms of the ground truth. This is not the case in unsupervised methods, since no ground truth is available.

Some of the properties in supervised learning (eg. presence of class labels) cannot obviously be transferred to outlier analysis. In other cases, analogous methods can be designed for the problem of outlier analysis. In the below, we discuss some common methods used for different supervised and unsupervised problems, and whether they can be transferred to the problem of outlier analysis:

- *Boosting:* Boosting [16] is a common technique used in classification. The idea is to focus on successively difficult portions of the data set in order to create models which can classify the data points in these portions more accurately, and then use the ensemble scores over all the components. A hold-out approach is used in order to determine the incorrectly classified instances for each portion of the data set. Such an approach clearly does not seem to be applicable to the unsupervised version of the problem because of the difficulty in computing the accuracy of the model on different data points in the absence of ground truth. On the other hand, since the supervised version of the problem (rare class detection) is a skewed classification problem, the boosting approach is applicable almost directly. A number of learners [13; 23] have been proposed for the supervised version of the outlier analysis problem. These classifiers have been shown to achieve significantly superior results because of the use of boosting. However, it is unlikely that an analogue for this method can be developed for a problem such as outlier analysis.

- *Bagging:* Bagging [9] is an approach which works with samples of the data, and combines the results from the different samples. The well known *feature* bagging approach for outlier analysis [17; 28] performs this step in a different way by bagging the features rather than bagging the points. Nevertheless, the approach is also applicable to the use of samples of data points (rather than dimensions) in order to perform the prediction. The key challenge in this case may arise from different data points being a part of different numbers of ensembles in the different cases. For many ensemble scores such as the *Maximum*, this can cause inadvertent bias in favoring data points which are sampled a larger number of times. This bias cannot be corrected, unless specific kinds of combination functions such as the *Average* are used.

- *Random Forests:* Random forests [8] are a method which use sets of decision trees on the training data, and compute the score as a function of these different components. While decision trees were not originally designed for the outlier analysis problem, it has been shown in [29] that the broad concept of decision trees can also be extended to outlier analysis by examining those paths with unusually short length, since the outlier regions tend to get isolated rather quickly. An ensemble of such trees is referred to as an *isolation forest* [29], and has been used effectively for making robust predictions about outliers.

- *Model Averaging and Combination:* This is one of the most common models used in ensemble analysis and is used both for the clustering and classification problems. In fact, the random forest method discussed above is a special case of this idea. In the context of the classification problem, many Bayesian methods [15] exist for the model combination process. Many of the recent models [25; 34] have focussed on creating a bucket of models from which the scores are combined through either averaging or using the maximum value. Even the parameter tuning methods used in many outlier analysis algorithms such as LOF and LOCI can be viewed to be drawn from this category. A related model is *stacking* [14; 38], in which the combination is performed in conjunction with model evaluation. This can sometimes be more difficult for unsupervised problems such as classification. Nevertheless, since stacking has been used for some unsupervised problems such as density estimation [37], it is possible that some of the techniques may be generalizable to outlier analysis, as long as an appropriate model for quantifying performance can be found.

- *Bucket of Models:* In this approach [39] a “hold-out” portion of the data set is used in order to decide the most appropriate model. The most appropriate model is one in which the highest accuracy is achieved in the held out data set. In essence, this approach can be viewed as a competition or bake-off contest between the different models. While this is easy to perform in supervised problems such as classification, it is much

Method	Model-Centered or Data-Centered	Sequential or Independent	Combination Function	Normalization
LOF Tuning [12]	Model	Independent	Max	Not Needed
LOCI Tuning [35]	Model	Independent	Max	Not Needed
Feature Bagging [28]	Data	Independent	Max/Avg	No
HICS [24]	Data	Independent	Selective Avg	No
Calib. Bagging [17]	Both	Independent	Max/Avg	Yes
OutRank [33]	Data	Independent	Harmonic Mean	No
Multiple Proclus [33]	Data	Independent	Harmonic Mean	No
Converting scores to probabilities [25]	Both	Independent	Max/Avg	Yes
Intrusion Bootstrap [6]	Data	Sequential	Last Component	Not Needed
OUTRES [30]	Data	Sequential	Product	No
Nguyen et al [34]	Both	Independent	Weighted Avg.	No
Isolation Forest [29]	Model	Independent	Expon. Avg.	Yes

Table 2: Characteristics of Outlier Ensemble Methods

more difficult for small-sample and unsupervised problems. No ground truth is available for evaluation in unsupervised problems. It is unlikely that a precise analogue of the method can be created for outlier analysis, since exact ground truth is not available for the evaluation process.

To summarize, we create a table of the different methods, and the different characteristics such as the type of ensemble, combination technique, or whether normalization is present. This is provided in Table 2.

6. CONCLUSIONS AND DISCUSSION

This paper provides an overview of the emerging area of outlier ensemble analysis, which has seen increasing attention in the literature in recent years. Many ensemble analysis methods in the outlier analysis literature are not recognized as such in a formal way. This paper provides an understanding of how these methods relate to other techniques used explicitly as ensembles in the literature. We provided different ways of categorizing the outlier analysis problems in the literature, such as independent or sequential ensembles, and data- or model-centered ensembles. We discussed the impact of different kinds of combination functions, and how these combination functions relate to different kinds of ensembles. The issue of choosing the right combination function is an important one, though it may depend upon the structure of the ensemble in the general case. We also provided a mapping of many current techniques in the literature to different kinds of ensembles. Finally, a discussion was provided on the feasibility of adapting ensemblar techniques from other data mining problems to outlier analysis. The area of ensemble analysis is poorly developed in the context of the outlier detection problem, as compared to other data mining problems such as clustering and classification. The reason for this is rooted in the greater difficulty of judging the quality of a component of the ensemble, as compared to other data mining problems such as classification. Many models such as stacking and boosting in other data mining problems require a crisply defined judgement of different ensemblar components on hold-out sets, which are not readily available in data mining problems such as outlier analysis. The outlier analysis problem suffers from the problem of small sample space as well as lack of ground truth (as in all unsupervised problems). The lack of ground truth implies that it is necessary to use the intermediate outputs of the algorithm (rather than concrete quality measures on hold-out sets), for making the combination decisions and ensemblar choices. These intermediate outputs may sometimes represent poor estimations of outlier scores. When combination decisions and ensemblar choices are made in an unsupervised way on an inherently small sample space problem such as outlier analysis, the likelihood and consequences of inappropriate choices can be high as compared to another unsupervised problem such as clustering, which does not have the small sample space issues.

While outlier detection is a challenging problem for ensemble analysis, the problems are not unsurmountable. It has become clear, from the results of numerous recent ensemble methods that such methods can lead to significant qualitative improvements. Therefore, ensemble analysis seems to be an emerging area, which can be a fruitful research direction for improving the quality of outlier detection algorithms.

7. REFERENCES

[1] C. C. Aggarwal. Outlier Analysis, *Springer*, 2013.

[2] C. C. Aggarwal, C. Reddy. Data Clustering: Algorithms and Applications, *CRC Press*, 2013.

[3] C. C. Aggarwal and P. S. Yu. Outlier Detection in High Dimensional Data, *ACM SIGMOD Conference*, 2001.

[4] C. C. Aggarwal, C. Procopiuc, J. Wolf, P. Yu, and J. Park. Fast Algorithms for Projected Clustering, *ACM SIGMOD Conference*, 1999.

[5] F. Angiulli, C. Pizzuti. Fast outlier detection in high dimensional spaces, *PKDD Conference*, 2002.

[6] D. Barbara, Y. Li, J. Couto, J.-L. Lin, and S. Jajodia. Bootstrapping a Data Mining Intrusion Detection System. *Symposium on Applied Computing*, 2003.

[7] S. Bickel, T. Scheffer. Multi-view clustering. *ICDM Conference*, 2004.

[8] L. Brieman. Random Forests. *Journal Machine Learning archive*, 45(1), pp. 5–32, 2001.

[9] L. Brieman. Bagging Predictors. *Machine Learning*, 24(2), pp. 123–140, 1996.

[10] V. Chandola, A. Banerjee, V. Kumar. Anomaly Detection: A Survey, *ACM Computing Surveys*, 2009.

[11] S. D. Bay and M. Schwabacher, Mining distance-based outliers in near linear time with randomization and a simple pruning rule, *KDD Conf.*, 2003.

[12] M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander. LOF: Identifying Density-based Local Outliers, *ACM SIGMOD Conference*, 2000.

[13] N. Chawla, A. Lazarevic, L. Hall, and K. Bowyer. SMOTEBoost: Improving prediction of the minority class in boosting, *PKDD*, pp. 107–119, 2003.

[14] B. Clarke, Bayes Model Averaging and Stacking when Model Approximation Error cannot be Ignored, *Journal of Machine Learning Research*, pp 683–712, 2003.

[15] P. Domingos. Bayesian Averaging of Classifiers and the Overfitting Problem. *ICML Conference*, 2000.

[16] Y. Freund, R. Schapire. A Decision-theoretic Generalization of Online Learning and Application to Boosting, *Computational Learning Theory*, 1995.

[17] J. Gao, P.-N. Tan. Converting output scores from outlier detection algorithms into probability estimates. *ICDM Conference*, 2006.

[18] Z. He, S. Deng and X. Xu. A Unified Subspace Outlier Ensemble Framework for Outlier Detection, *Advances in Web Age Information Management*, 2005.

[19] D. Hawkins. Identification of Outliers, *Chapman and Hall*, 1980.

[20] A. Hinneburg, D. Keim, and M. Wawryniuk. Hd-eye: Visual mining of high-dimensional data. *IEEE Computer Graphics and Applications*, 19:22–31, 1999.

[21] W. Jin, A. Tung, and J. Han. Mining top-n local outliers in large databases, *ACM KDD Conference*, 2001.

[22] T. Johnson, I. Kwok, and R. Ng. Fast computation of 2-dimensional depth contours. *ACM KDD Conference*, 1998.

[23] M. Joshi, V. Kumar, and R. Agarwal. Evaluating Boosting Algorithms to Classify Rare Classes: Comparison and Improvements. *ICDM Conference*, pp. 257–264, 2001.

[24] F. Keller, E. Muller, K. Bohm. HiCS: High-Contrast Subspaces for Density-based Outlier Ranking, *IEEE ICDE Conference*, 2012.

[25] H. Kriegel, P. Kroger, E. Schubert, and A. Zimek. Interpreting and Unifying Outlier Scores. *SDM Conference*, 2011.

[26] E. Knorr, and R. Ng. Algorithms for Mining Distance-based Outliers in Large Datasets. *VLDB Conference*, 1998.

[27] E. Knorr, and R. Ng. Finding Intensional Knowledge of Distance-Based Outliers. *VLDB Conference*, 1999.

[28] A. Lazarevic, and V. Kumar. Feature Bagging for Outlier Detection, *ACM KDD Conference*, 2005.

[29] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation Forest. *ICDM Conference*, 2008.

[30] E. Muller, M. Schiffer, and T. Seidl. Statistical Selection of Relevant Subspace Projections for Outlier Ranking. *ICDE Conference*, pp, 434–445, 2011.

[31] E. Muller, S. Gunnemann, I. Farber, and T. Seidl, Discovering multiple clustering solutions: Grouping objects in different views of the data, *ICDM Conference*, 2010.

[32] E. Muller, S. Gunnemann, T. Seidl, and I. Farber. Tutorial: Discovering Multiple Clustering Solutions Grouping Objects in Different Views of the Data. *ICDE Conference*, 2012.

[33] E. Muller, I. Assent, P. Iglesias, Y. Mulle, and K. Bohm. Outlier Ranking via Subspace Analysis in Multiple Views of the Data, *ICDM Conference*, 2012.

[34] H. Nguyen, H. Ang, and V. Gopalakrishnan. Mining ensembles of heterogeneous detectors on random subspaces, *DASFAA*, 2010.

[35] S. Papadimitriou, H. Kitagawa, P. Gibbons, and C. Faloutsos, LOCI: Fast outlier detection using the local correlation integral, *ICDE Conference*, 2003.

[36] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient Algorithms for Mining Outliers from Large Data Sets. *ACM SIGMOD Conference*, pp. 427–438, 2000.

[37] P. Smyth and D. Wolpert. Linearly Combining Density Estimators via Stacking, *Machine Learning Journal*, 36, pp. 59–83, 1999.

[38] D. Wolpert. Stacked Generalization, *Neural Networks*, 5(2), pp. 241–259, 1992.

[39] B. Zenko. Is Combining Classifiers Better than Selecting the Best One, *Machine Learning*, pp. 255–273, 2004.